Toward software updates in IoT environments

Jean-Baptiste Trystram

Toward Software Updates in IoT Environments: Why Existing P2P Systems are not Enough

Jean-Baptiste Trystram
Univ Lyon, INSA Lyon, CITI Red Hat Inc.
F-69621 Villeurbanne, France 92800 Puteaux, France
jbtrystram@redhat.com

ABSTRACT

The number of connected devices around us is growing, and their embedded software is increasing in complexity. Applications need to be managed and updated throughout their life-cycle. Modern software orchestrators and management systems are mostly centralised and expensive to scale to very large systems. We propose a decentralised approach to distribute software updates to Internet of Things (IoT) devices using Peer-to-Peer (P2P) transfers. We seek to improve the efficiency of software update propagation by adapting system behaviour to IoT constraints, such as device heterogeneity, unreliable network connectivity and applications specifics.

CCS CONCEPTS

• **Networks → Peer-to-peer protocols**; *Sensor networks*; *Peer-to-peer networks*; *Mobile ad hoc networks*; • **Computer systems organization → Peer-to-peer architectures**; • **Software and its engineering → Software maintenance tools**;

ACM Reference Format:
Jean-Baptiste Trystram. 2017. Toward Software Updates in IoT Environments: Why Existing P2P Systems are not Enough. In *Middleware Doct Symposium 17: Middleware Doct Symposium 17: Proceedings of the Doctoral Symposium of the 18th International Middleware Conference, December 11–15, 2017, Las Vegas, NV, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3152688.3152694

1 INTRODUCTION

The IoT landscape will include billions[1] of devices deployed in various environments such as smart industries and cities, connected cars and healthcare. As these devices get new features, they will run increasingly complex software. Like all software, it needs to be patched for feature improvements and security fixes. A large number of these devices puts a strain on the traditionally centralised software deployment schemes as modern networks cannot cope with intensive M2M usage [3]. Additionally, centralised techniques are designed for static environments while IoT devices have unpredictable network connectivity and heterogeneous hardware. P2P systems handle these constraints but usually deal only with scalability and resiliency issues, so we propose to extend these techniques to take account of energy usage and IoT application-specific requirements.

[1]Gartner's forecast for IoT, Feb. 2017. http://www.gartner.com/newsroom/id/3598917

Middleware Doct Symposium 17, December 11–15, 2017, Las Vegas, NV, USA

ACM ISBN 978-1-4503-5199-7/17/12.
https://doi.org/10.1145/3152688.3152694

2 RELATED WORK

Service discovery in distributed systems has been widely studied [8, 12]. Such techniques locate a given service or file within the system but do not offer ways to access them. Moreover, some do not scale to large and disconnected networks. For example, Distributed Hash Table (DHT) requires connectivity with the node storing the desired entry [8]. Additionally, there is no support for versioning: services or files cannot co-exist with different versions; they are treated as unrelated.

Software management systems are designed to handle versioning and large systems, but they are mostly centralised, thus requiring a permanent connection with the orchestrator. De-centralised flavours do exist, but still, rely on a central server to coordinate peers [2].

Today's software distribution mechanisms are almost exclusively based on a client-server model. P2P software distribution exists but is mainly focused on offloading work from the central server [1]. Peer locality [6] reduces inter-ISP data traffic but relies on a central service.

Ad-hoc networks are a conducive environment for P2P applications [5, 7]. While MANETs share similar constraints to IoT (mobility, energy limits, unstable connectivity) the scale is different. Indeed, MANET networks are fairly small, and all the nodes are regularly connected.

Lastly, the network usage related to software updates is non-negligible [9]. Modern centralised mobile infrastructures cannot scale to a large amount of data per user [11] nor to a vast number of devices [3].

3 INTENDED APPROACH

In this section, we lay down the key features of a new software update dissemination system that fit IoT constraints and adapts to applications specifics. First, we describe how a device discovers available content. We then argue how constraints imply the need of downloading from multiple sources. Next, we introduce piece selection strategies and their impact on the system's efficiency. Security aspects are then discussed, and how metrics collection improves reliability.

Content discovery. A large number of unpredictable devices implies that a fully connected graph is not realistic. Thus, a DHT or a tracker-based system is not efficient. We assume nodes will spontaneously initiate ad-hoc connections with their neighbours when possible. To make use of these connections, devices need to know whether they have a shared interest in certain resources (e.g. software updates).

Announce-based protocols have proven reliable in networks with inconsistent connectivity (e.g. vehicular networks), providing efficient epidemic or gossip-style dissemination [4]. Using this announcement, devices can exchange details about the software they run and share, together with, their associated versions. Devices can initiate transfer of packages included in announcements received.

Multiple parts download. Spontaneous connections between mobile devices may be too short to exchange a full package between two devices. To cope with this issue, files can be divided into small chunks. Block usage allows to resume an interrupted download, or get the remaining parts of the file from other peers.

Piece selection strategy. The main challenge is to design a piece selection algorithm that efficiently distributes files throughout the network. There are several block selection approaches to improve file dissemination. BitTorrent chooses blocks randomly or picks the rarest first. Specific strategies could make use of the dynamic properties of devices, such as mobility and social interactions. Social patterns help to predict how devices will meet [13]. Adding the last devices met in the announcement message facilitate this approach.

Application dependencies require consideration, as updating some software may be impossible without up-to-date dependencies. In this case, dependencies must be downloaded first. The announcement protocol should express such constraints. This is critical for low memory devices: they may not be able to hold all of the update packages in memory alongside the running application. It may be necessary to process updates as they download them, in which case the downloading order cannot be random.

Security. To avoid a malicious device tampering with the data before sharing it, cryptographic signatures can be used. As a central server is not always reachable, downloads should contain a hash of the pieces, signed with the publisher's keys (as in Android and iOS applications).

Application-centric. A non content-agnostic protocol would provide improved features. The addition of metadata along packages allow to adjust convergence and consistency accordingly (e.g. a critical security fix needs a faster deployment). Such metadata could also host statistics on application deployment results in a distributed manner. Devices may use feedbacks from their peers to avoid installation errors or rollback to previous versions.

Accomplished work

Initial work has been done modifying the PeerSim[2] simulator to reproduce a more realistic IoT scenario. We added dynamic properties to the simulation engine: nodes mobility, free-riders, network bandwidth heterogeneity and limited energy. Nodes only pick peers within a given range when creating the neighbour table. The next step is to include an advanced mobility model; at the time of writing, we use a simple random walk.

4 EVALUATION

Several mobility data sets can be used to test our solution with realistic data. The TAPAS Cologne[3] project logged vehicular traffic within the city of Cologne for 24h. The San Francisco cab data set[4] contains mobility trace of 500 cabs for 30 days. Those provide data representative of mobility patterns that we will use to measure how quickly an update propagates.

The primary performance metric being considered is the network volume overhead: how much more network data is exchanged compared to a simpler client-server download. The overhead for a single device will be measured from which the overhead for the full network will be extrapolated. Values will be compared to a standard P2P file sharing system like BitTorrent. These metrics should be reported to the time needed to get the update on every device. Additionally, a Markov model can describe the system behaviour in its stationary state. This approach and have been used previously with accurate results [10, 14].

5 CONCLUSION

P2P dissemination of software updates in IoT systems will lower infrastructure load while increasing efficiency and reliability in cases where strong consistency and quick convergence is not needed. The update process should also satisfy IoT constraints: after the download, a safe installation is critical. A reliable rollback system will prevent the device from entering a blocking state, as manual intervention may be costly or impossible for some devices.

ACKNOWLEDGMENTS

I want to thank my advisers Julien Ponge, Frédéric Le Mouël, David Ingham and Mark Little for their guidance and support during this first year of PhD. This work is supported by Red Hat Inc.

REFERENCES

[1] S. Abiteboul, I. Dar, R. Pop, G. Vasile, D. Vodislav, and N. Preda. 2007. Large Scale P2P Distribution of Open-Source Software. In *Proc. of VLDB.*
[2] J.P. Arcangeli, R. Boujbel, and S. Leriche. 2015. Automatic deployment of distributed software systems: Definitions and state of the art. *Journal of Systems and Software* (2015).
[3] S. Cherkaoui, I. Keskes, H. Rivano, and R. Stanica. 2016. LTE-A Random Access Channel Capacity Evaluation for M2M Communications. In *Proc. of IFIP Wireless Days.*
[4] A. Das, I. Gupta, and A. Motivala. 2002. SWIM: Scalable Weakly-consistent Infection-style Process Group Membership Protocol. In *Proc. of DSN.*
[5] G. Ding and B. Bhargava. 2004. Peer-to-peer File-sharing over Mobile Ad hoc Networks. In *Proc. of PerCom Workshops.*
[6] C. Gkantsidis, T. Karagiannis, P. Rodriguez, and M. Vojnovic. 2006. Planet scale software updates. In *Proc of SIGCOMM.*
[7] S. Goel, M. Singh, D. Xu, and B. Li. 2002. Efficient Peer-to-Peer Data Dissemination in Mobile Ad-Hoc Networks. In *Proc. of ICPP Workshops.*
[8] E. Meshkova, J. Riihijärvi, M. Petrova, and P. Mähönen. 2008. A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. *Computer Networks* (2008).
[9] M. Orehek and A. Zugenmaier. 2016. Updates in IoT are more than just one iota. In *IAB IoT Software Update Workshop.*
[10] D. Qiu and R. Srikant. 2004. Modeling and performance analysis of BitTorrent-like peer-to-peer networks. In *Proc. of SIGCOMM.*
[11] H. Rivano, I. Augé-Blum, W. Bechkit, K. Boussetta, M. Fiore, R. Stanica, and F. Valois. 2014. Wireless Access Networks for Smart Cities. In *Social, Economic, Environmental Sustainability in the Development of Smart Cities.*
[12] F. Sailhan and V. Issarny. 2005. Scalable Service Discovery for MANET. In *Proc. of PerCom.*
[13] P. Vaz de Melo, A. Carneiro Viana, M. Fiore, K. Jaffrès-Runser, F. Le Mouël, L. Loureiro, A. Addepalli, and G. Chen. 2015. RECAST: Telling apart social and random relationships in dynamic networks. *Performance Evaluation* (2015).
[14] L. Zou and M. Ammar. 2003. A File-Centric Model for Peer-to-Peer File Sharing Systems. In *Proc. of ICNP.*

[2] http://peersim.sourceforge.net
[3] http://sumo.dlr.de/wiki/Data/Scenarios/TAPASCologne
[4] http://crawdad.org/epfl/mobility/20090224